COLLEGE OF ENGINEERING / DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Q-Learning for Driving on Infinite Road

Hao Chen, Jianzong Pi

Abstract

The autonomous vehicles will consume large amounts of streaming data from nearby vehicles, infrastructure, and cloud to make decisions. The goal of the project is to adaptively change the velocity of a connected autonomous vehicle to minimize the fuel consumption of the vehicle. The vehicle will make real time decisions based on real-time information from V2X and V2V connectivity keeping in mind the constraints imposed by the powertrain of the vehicle. We aim at solving this problem using reinforcement learning algorithm. The ultimate goal of the project is to design and run very large scale reinforcement learning algorithm for devising fuel efficient driving strategies.

INTRODUCTION

Our group started the first set of reinforcement learning implementation for a car on an infinite road. The goal is to train a vehicle to driving itself in the center of the road and test the first simple Q-learning algorithm as a stepping stone for more complex simulation to be done in the project. Currently, we setup the environment to be a simple straight roads with one car only. The Q-learning algorithm is designed to find a driving policy to keep the car at or steering toward the center of the road. We created this scenario just to build and test our Q-learning framework, as well as exploring with different function approximators.

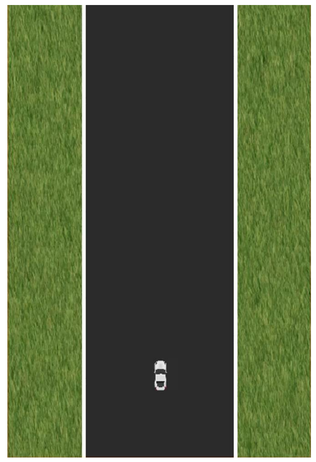

Figure 1: Simulation Environment

SETUP

We started with Q-table, an entry-level function approximator, to record the reward value for a given state and action. A Q-table is a matrix $Q \coloneqq Q(S_t, A_t)$ in which every row correspond to a state S_t, the distance from the center of the road sampled at each meter, and every column correspond to an action A_t, the steering angle ranging from -0.07 to 0.07 radian with 21 sample points. For this problem, we chose the reward function as:

$$\text{reward}=\begin{cases}50-S_t^2 & if\ car\ is\ on\ the\ road\\ -2000 & otherwise\end{cases}$$

This reward function provides the car with maximum reward when the car is at the center of the road ($S_t = 0$) and provides a penalty for going off the road.

ALGORITHM

The Q-learning algorithm with Q-table method is given as

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(S_t, A_t)]$$

We pick the learning rate $\alpha = 0.01$ and the discount factor $\gamma = 0.9$. We ran this update algorithm in every iteration in which 90% of the time the car takes the optimal action defined as, $A_t^* = argmax_a Q(S_t, a)$ and any other random action 10% of the time to explore the other possible combinations of S_t and A_t. Each episode stop at 10-second time stamp or whenever the car exits the road. The pseudo code for Q learning is given below:

Initialize $Q(s, a)$ arbitrarily
Repeat (for each episode):
  Initialize s
  Repeat (for each step of episode):
    Choose a from s using policy derived from Q
    Take action a, observe a, s'
    $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma \max_{a'} Q(s', a') - Q(s,a)\right]$
    $s \leftarrow s'$
  until s is terminal

Qtable	Action						
State	0	0	0	0	0	0	0
	...	...	...	...	...	...	...
	-2.3	-1.9	-2.6	-8.9	5.4	6.0	-2.2
	...	...	...	...	...	...	...
	9.6	4.2	12.9	29	32	3.5	9.8

Figure 2: Example Q Table

RESULTS

The figure below is generated by selecting the action that yields the maximum reward at each state. The y axis is the steering angle in radian and the x axis is horizontal position in meters. Horizontal position is ranging from 0 to 20 and 10 is the middle of the road. When the position is blew 10, vehicle is on the left side of the road, and turning right yields the maximum reward and vice versa. This figure shows the car has learned that keeping its centered in the road is the best decision to make under the scenario defined by the reward function.

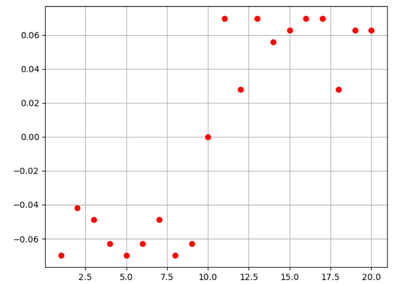

Figure 3: Steering angle vs. Horizontal position of the trained policy

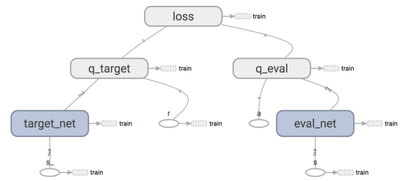

Figure 4: Example Neural Networks

FUTURE WORK

Although Q-table is straightforward and easy for us to code, its inherent spatial complexity will be uncontrollable when problem gets sophisticated. In connected autonomous cars, the number of states will grow exponentially, and therefore, Q-table is no longer a viable option. We are exploring the use of other function approximators such as neural networks to predict the Q-value. Currently, a new version of code is under development to use neural networks in replacement of the Q-table. This provide several advantages over the Q-table, including better training potential, more discretized states, faster training time, etc. In addition, we will also modify our code for parallel computing to achieve higher training efficiency while running more complexed scenarios.

Our ultimate goal is to optimize the fuel consumption for multiple connected autonomous vehicles. Therefore, teaching them to driving themselves would be the first step. In the near future, we will integrate real-world engine model into our algorithm for the computation of the fuel consumption to yield meaningful results.

REFERENCE

1. R. S. Sutton and A. Barto, *Reinforcement learning: an introduction*. Cambridge, MA: The MIT Press, 2018.

ACKNOWLEDGEMENTS

We gratefully acknowledge NSF CRII Award 1565487 and ARPA-E NEXTCAR Award for supporting this research.